SPACE SCIENCES LABORATORY

N65 17946
(ACCESSION NUMBER)
52
(PAGES)
CR-60952
(NASA CR OR TMX OR AD NUMBER)
FACILITY FORM 602

(THRU)
1
(CODE)
08
(CATEGORY)

UNPUBLISHED

UNIVERSITY OF CALIFORNIA

BERKELEY, CALIFORNIA

GPO PRICE $ ______

OTS PRICE(S) $ ______

Hard copy (HC) $3.00

Microfiche (MF) $0.50

A COMPARISON AMONG EIGHT KNOWN OPTIMIZING PROCEDURES

by

Alberto Leon

Internal Working Paper No. 20
August 1964

Space Sciences Laboratory
University of California
Berkeley

This research was supported in part by the National Aeronautics and Space Administration under General Grant #NSG-243-62 under the University of California. [redacted]

TABLE OF CONTENTS

INTRODUCTION

It is intended here to present some results of our research on optimization techniques. Eight known optimum seeking methods are used to optimize five simple two-variable unconstrained functions. Four of the problems are presented and analyzed by Witte and Holst[1] and the other one by Beale.[2] The type of optimization process reported here is that of locating a set of values of a set of variables that yields either a minimum or a maximum value for a function given in algebraic form.

Each one of the optimizing techniques is programmed in FORTRAN II language for IBM 709/7090 computers. These programs will be referred to as CODES from here on.

This instructive exercise is carried out to acquire knowledge on the different operational characteristics of the computer codes. Familiarity with the program parameters of each one of the codes and understanding of their internal stopping rules is required to introduce the necessary changes for them to be used in connection with GROPE. GROPE is a Universal Adaptive Code for Optimization developed by Professor Merrill M. Flood and the author[3, 4] at the University of California, Berkeley. A Universal

---

[1]Witte, Bruno F., and William R. Holst, "Two New Direct Minimum Search Procedures for Functions of Several Variables," submitted for presentation at the 1964 Spring Joint Computer Conference in Washington, D. C. (27 pages).

[2]Beale, E. M. L., "On an Iterative Method for Finding a Local Minimum of a Function of More Than One Variable," Technical Report No. 25, Statistical Techniques Research Group, Princeton University, November 1958 (44 pages).

[3]Flood, Merrill M., and Alberto Leon, "A Universal Adaptive Code for Optimization (GROPE)," Space Sciences Laboratory, University of California, Internal Working Paper No. 19, August 1964 (100 pages).

[4]Leon, Alberto, "Steps Toward a Universal Adaptive Code for Optimization (GROPE)," Space Sciences Laboratory, University of California, Internal Working Paper No. 11, April 1964 (26 pages).

Adaptive Code for Optimization, as we see it, is a general code which selects adaptively and sequentially among a group of several optimizing codes as each problem calculation progresses.

It is also found of interest to compare the behavior of these techniques under identical conditions, that is, with the same set of problems under the same computing system (University of California, Berkeley, Executive System).

We do not attempt a detailed description of each technique and the reader is referred to the proper references. Our objective here is to present some results rather than extensive descriptions or analysis of the codes.

THE OPTIMIZING TECHNIQUES

The optimization codes we are dealing with are VARMINT, MINFUN, STEP, LOOK, BEST UNIVAR, ITERATED PARTAN, CONTINUED PARTÁN, and a version of the STEEPEST DESCENT method contained in the PARTAN code.

The eight techniques can be classified into two broad categories: techniques based on conventional mathematical methods and techniques of the DIRECT SEARCH type.

VARMINT, MINFUN, STEP, ITERATED PARTAN, CONTINUED PARTAN and STEEPEST DESCENT belong to the group of conventional mathematical methods. These techniques use in some way or another the gradient of the function to be optimized and so require the analytical or numerical evaluation of the partial derivatives of that function. The gradient brings the idea of the direction of fastest improvement toward a solution (either ascending or descending), obviously of great significance. It is enough to say here that the gradient vector points in the direction in which the function increases or decreases most rapidly and its length is the rate of increase or decrease in that direction.

BEST UNIVAR and LOOK are representatives of the DIRECT SEARCH type of optimizing procedures. Hook and Jeeves[5] have written that: "DIRECT SEARCH is just sequential examination of trial solutions. Each trial solution is compared with the 'best' obtained up to that time, and there is a strategy for determining (as a function of earlier results) what the next trial solution will be."

Next, we described briefly some of the relevant aspects of the eight codes we are using here.

1. VARMINT (VARIABLES METRIC METHOD FOR MINIMIZATION):[6,7,8]

Davidon says: "This is a method for determining numerically local minima of differentiable functions of several variables. In the process of locating each minimum, a matrix which characterizes the behavior of the function about the minimum is determined. For a region in which the function depends quadratically on the variables, no more than N iterations are required, where N is the number of variables. By suitable choice of starting values and without modification of the procedure, linear constraints can be imposed upon the variables."

---

[5]Hooke, R. and T. A. Jeeves, "Direct Search Solution of Numerical and Statistical Problems," J. Assoc. Computing Mach., Vol. 8, 1961, 212-229.

[6]Davidon, C. William, "Variable Metric Method for Minimization," Argonne National Laboratory, ANL-5990, November, 1959 (21 pages).

[7]Stevens, D. F., "Instructions for the User of VARMINT, Deck ZOEOZO13, Lawrence Radiation Laboratory, University of California, Berkeley, June 1961 (18 pages).

[8]Fend, F. A., and C. B. Chandler, "Numerical Optimization for Multidimensional Problems," General Electric, General Engineering Laboratory, Report No. 61 GL78, March 1961 (47 pages).

Fend and Chandler point out that, "Gradient methods basically analyze the changes in slope (i.e., vector components of the gradient) corresponding to changes in the position of the trial point. They attempt to drive the components of the slope to zero and thus obtain the location of the optimum point which is sought. Davidon's method systematically varies the metric which specifies the change in vector components of the gradient corresponding to changes in the location of the best point. In this respect it may be characterized as an adaptive procedure. It has a further advantage in that it uses an interpretative procedure once the optimum point is bracketed."

In the neighborhood of any one point the second derivatives of the function to be optimized, f(X)*, specify a linear mapping of changes in position, <u>dX</u>, onto changes in gradient <u>d∇</u>. These changes are expressed for a change in the ith derivative, for example, as

$$d\left(\frac{\partial f}{\partial X_i}\right) = \sum_{j=1}^{N} \frac{\partial^2 f}{\partial x^i \; x^j} dX^j$$

$$d\left(\frac{\partial f}{\partial X_i}\right) = 1H^{ij} dX^j$$

where H^{ij} is the Hessian matrix. As we know, the optimum point will require that $\frac{\partial f}{\partial X_i}$ vanishes and so the desired change in X_i (under the assumption that the Hessian matrix is constant) will be

$$dX^{\sqrt{}} = \left\| H^{ij} \right\|^{-1} \left[- \frac{\partial f}{\partial X_i} \right]$$

In general H^{ij} does not remain fixed and here lies the important contribution of VARMINT: correction of the Hessian matrix from iteration to iteration. This idea was mentioned by Crockett and Chernoff[9] while discussing

---

*f denotes the function to be optimized.

[9]Crockett, J. B., and H. Chernoff, "Gradient Methods of Maximization," <u>Pacific J. of Math</u>., 5, 1955, 33-50.

the differences between the Newton method and the gradient methods.

The matrix H can be visualized as an error matrix and must be a positive definite matrix. A suggested initial value for H is

$$H^{ii} = (\delta X_i)^2$$

$$H^{ij} = 0 \quad \text{for } i \neq j$$

where δX_i is estimated error in X_i. In the absence of a better estimate, H may be taken to be the identity matrix of order N.

We use the version of VARMINT available at the Lawrence Radiation Laboratory, the University of California, Berkeley (Deck ZOEOZO13-FORTRAN II).

2. <u>MINFUN</u> (A GENERAL MINIMIZING ROUTINE):

Humphrey says[10]: "Briefly, the program is a FORTRAN control routine and two subroutines which are designed to be used with a function subroutine to be coded by the user. This group of programs uses the ravine stepping procedure to either explore the 'space' of the independent variables near the minimum or seek the actual set of variables at the minimum (at the option of the user). Provisions have been included to allow exclusion of regions of the variable space from the allowed steps."

The whole operation of MINFUN can be made clear by considering a hypothetical function f of two variables (x, y). Figure No. 1 in Appendix B shows the schematic representation of the optimization process. At point 0, the initial point, the starting direction is taken as being along the gradient. A step is taken transverse to the line 0-1 from the point 1 to point 2. At point 2 the function is evaluated. Using the information available at point 1 and 2, a minimum is predicted along line 1-2 at point 3. The

---

[10]Humphrey, W. E., "A General Minimizing Routine-Minfun," Lawrence Radiation Laboratory, University of California, Berkeley, Internal Memorandum, September 1962 (9 pages).

function is calculated at point 3 to verify the minimum at that point. To complete the cycle, a step is now taken along the line 0-3 to a point 1' and the operation repeats as described at point 1.

We are indebted to Mr. W. E. Humphrey for a copy of the FORTRAN II deck of his program as well as for fruitful conversations concerning the use of MINFUN.

3. STEP (AN EXTREMUM LOCATING ALGORITHM):

The procedure used by STEP is designed to circumvent the existence of local cols (in the surface which is generated by the function) which point in directions other than that of the minimum. When such cols exist, the subroutine uses two points along the spine of the col for extrapolation (in the direction of descent) to a point from which is sought the next spinal point. If the minimum appears to be overshot, then an interpolation takes place. Following this, probing parameters are scaled down, and the whole procedure is iterated until either convergence occurs or the procedure exceeds the limit on the number of iterations.

Baer[11] has written that: "Roughly put, the algorithm consists of using alternatively two procedures: EXPLORING and HOMING. Exploring consists of generating a sequence of restricted minima along the spine of the valley of the surface generated by the junction. Homing consists of interpolation between appropriate restricted minima when there is an indication that the neighborhood of the required minimum has been overshot.

The efficiency of the procedure lies in the mode of generation of the restricted minima. Having obtained more than one of these, one generates the next by extrapolation (an appreciable distance) in the direction of the

[11] Baer, Robert M., "Note on an Extremum Locating Algorithm," The Computer Journal, Vol. 5, No. 3.

vector difference of the preceding two, and then relying on the gradient. Except for the first in this sequence of restricted minima, no great care need be taken in their determination, inasmuch as they need not be exact."

It is interesting to add what Baer means by restricted minimum. "If α, β are taken to be fixed vectors, and if t is a (real-valued) scaler, then the minimum (with respect to t) of $f(\alpha + t\beta)$ will be called a RESTRICTED MINIMUM."

We used a FORTRAN II version of STEP available through the IBM Share System. We are indebted to Dr. R. M. Baer of the Computing Center, University of California, Berkeley, for helpful instructions to work properly with his code.

4. STEEPEST DESCENT

There are many codes using in different ways the steepest descent (or ascent) ideas. We apply here a straightforward steepest descent procedure available in PARTAN (described in a subsequent section).

This version of the steepest descent method may be called optimum gradient because it locates the optimum in the gradient direction at each point. The code works as follows:[12] (See Figure No. 2 in Appendix B.)

(i) Determine the direction of the gradient at the starting point P_o.

(ii) Locate the minimum on this "steepest descent" path; designate this point as P_2.

(iii) Determine the direction of the gradient at P_2.

(iv) Locate the minimum on this "steepest descent" path. Designate this point as P_3.

---

[12] Doerfler, T. E., "PARTAN, Minimization by Method of Parallel Tangents," Iowa State University, April 1964, Internal Memorandum (7 pages).

(v) Continue this procedure to P_N.

"A simple algorithm using cubic interpolation is employed to estimate the minimum on any line $x + \lambda s$, where x is the origin of the line, s is the vector determining the direction, and λ is the step-size parameter to be estimated."

It is interesting to notice that[13] in principle the steepest descent method will not reach the optimum in a finite number of steps because the steps shorten as the point is approached. However, the optimum can be approached as closely as desired, and if the starting point is not too near the major axis the neighborhood of the optimum is attained rapidly.

5. ITERATED PARTAN

The general PARTAN code includes the version of steepest descent described previously together with two variations of the PARALLEL TANGENTS (PARTAN) technique as presented by B. V. Shah *et al.*[14] Both variations of PARTAN look for some sort of acceleration of the steepest descent search. This is an attempt to reduce to a finite number the "infinite" number of steps required to reach the optimum by means of the steepest descent procedure.

The authors of PARTAN say that in the two algorithms one proceeds to optima of f on successive straight lines. The path directions are alternately determined by positions of points already reached or by certain gradient directions. They also say that all the theoretical results concern the "ideal" case, meaning by ideal:

---

[13] Wilde, D. J., Optimum Seeking Methods. Prentice-Hall, Inc., 1964.

[14] Shah, B. V., R. J. Buehler and O. Kempthorne, "Some Algorithms for Minimizing an Observable Function," Journal Soc. Ind. Appl. Math., Vol. 12, No. 1, March 1964, 74-92.

(1) f is quadratic;

(2) f and its gradient direction can be determined without error at any specified point;

(3) On any given line, the point at which f is an optimum can be determined without error.

In the absence of error the procedure converges exactly to the optimum in (2N - 1) steps for a quadratic function.

ITERATED PARTAN operates in the following way (Figure No. 3, Appendix B):

(i) Connect P_1 and P_3 and locate the minimum on this extended line. Designate this point as P_4.

(ii) From here on re-do the steps involved in the steepest descent process plus the previous one using P_4 as the starting point.

6. <u>CONTINUED PARTAN</u>

The so-called CONTINUED PARTAN, as was said before, is a variation of the previous one and it involves the following steps (Figure 4, Appendix B):[15]

(i) Determine the direction of the gradient at P_4;

(ii) Locate the minimum on this steepest descent path. Designate this point as P_5;

(iii) Connect P_2 and P_5 and locate the minimum on this line. Designate this point as P_6;

(iv) Repeat the previous steps until obtaining P_N; "always taking a 'steepest descent' direction at P_{2j}, $j = 2, 3, \ldots$ and connecting P_{2j-2} and P_{2j+1}, $j = 2, 3, \ldots$ for the PARTAN acceleration step."

---

[15] See footnote 12, page 7.

We are indebted to Dr. O. Kempthorne and Mr. Thomas E. Doerfler both from the Statistical Laboratory of Iowa State University for a copy of the FORTRAN II deck of PARTAN and its operating instructions.

7. LOOK

LOOK is fully described in the reference in footnote 5 on page 3. It may be described briefly as follows:[16]

"1. Initialization. A starting point for the search is calculated[17] and stored.

"2. Exploratory search. Various moves are made to determine a desirable direction for the search. Any move which is better than the reference value is kept and becomes the new reference value. On the initial entry or whenever the exploratory search is not immediately preceded by a pattern move, the reference value is the last base point. Following a pattern move, the reference is the value at the end of the pattern move.

"3. Success? If the best value found for the function during the exploratory search is better than its value at the last base point, a new base point is established. Otherwise, the last base point is restored.

"4. Save base point and make Pattern Move. The latest functional value replaces the previous value and the corresponding values of the independent variables do likewise. This establishes a new base point. The pattern

---

[16] Wood, C. F., "Recent Developments in 'Direct Search' Techniques," Westinghouse Research Report 62-159-522-R1.

[17] Or given as Data.

move is generated by moving each independent variable away from the latest base point value by an amount equal to the difference between the old and new base point values. A pattern move is always followed immediately by an exploratory search.

"5. Restore last Base Point. The independent variables are set at the values corresponding to the last base point. The functional value for the same point becomes the initial reference for testing the individual moves of the exploratory search.

"6. Had Pattern Move just been made. If the exploratory search preceding the failure was itself preceded by a pattern move, perform another exploratory search. Otherwise, check for search completion.

"7. Can step size be reduced? If the step sizes for all the independent variables are at their minima, the search is complete. Otherwise, reduce step size and perform another exploratory search."

As we see the final termination of the search is made when the step size is sufficiently small to ensure that the optimum has been closely approximated. In any case, the step size must be kept above a practical limit imposed by the means of computation. The search is stopped when two conditions occur at the same time, namely (1) the step size is at minimum and (2) the forward and reverse moves of all independent variables fail following a base point test failure.

As Hooke and Jeeves say, "In practice, pattern search has proved particularly successful in locating minima on hyper surfaces which contain 'sharp valleys'. On such surfaces classical techniques behave badly and can only be induced to approach the minimum slowly."

We are indebted to Mr. C. F. Wood for a copy of the deck of LOOK's original FORTRAN II code.

8. BEST UNIVAR

This Direct Search Code uses one of the many possible strategies that might be employed to determine subsequent trials as a function of previous results.

BEST UNIVAR is fully described together with numerical examples in two papers written jointly by Professor Merrill M. Flood and the author.[18,19] BEST UNIVAR is available in FORTRAN II for IBM 709/7090 computers operated either under the University of Michigan or the University of California Executive Systems. Changes were introduced recently and the code is also available now in FORTRAN IV for IBM 7090/7094 computers processed by the FORTRAN IV compiler, and 7090/7094 IBJOB Processor Component.

BEST UNIVAR may be described very briefly as follows:

1. Initialization. The optimization process is initiated by picking up, as the starting point, an arbitrary point inside the operating space.
2. Order of analysis. Once the function has been evaluated at the starting point, the independent variables to be changed are changed in an order selected initially by the experimenter.

---

[18] Flood, Merrill M. and Alberto Leon, "A Direct Search Code for the Estimation of Parameters in Stochastic Learning Models," Preprint 109, Mental Health Research Institute, The University of Michigan, May 1963 (63 pages).

[19] Flood, Merrill M. and Alberto Leon, "A Generalized Direct Search Code for Optimization," Preprint 129, Mental Health Research Institute, The University of Michigan, June 1964 (64 pages).

3. One-at-a-time search. After deciding upon the order in which to search the one-at-a-time search is initiated. Let X_i be the first variable under study; this variable is incremented by an amount Δ_i, holding the other variables at their initial values. If the functional value at this point is better than the one at the preceding point there is some reason for trying further in the same direction. A larger step size is now used, taken equal to $\lambda_i\Delta_i$ (where $\lambda_i > 1$), and if a better functional value (comparing against the immediately previous one) is obtained, a step of length $\lambda_i^2\Delta_i$ is used next. We continue in the same direction by powers of λ_i until no further improvement is obtained. Assume that step $\lambda_i^{h+1}\Delta_i$ was the first unsuccessful one; in this case the preceding base point is kept, namely the one obtained by step $\lambda_i^h\Delta_i$ and a new sequence is started from this point with initial step size equal to Δ_i following the same scheme as before. If a step of Δ_i in the positive direction does not bring a better point, then a step of length Δ_i in the negative direction is tried; if this happens to be a successful step, the $\lambda_i\Delta_i$ is tried in the same negative direction continuing in the same fashion as was done in the positive direction. Finally, we reach a point where no improvement is obtained by moving variable X_i either Δ_i or $-\Delta_i$; this point is considered to be the best temporarily for variable X_i. After the best point in the X_i direction is found the second variable in the list is ready to be analyzed. The process is repeated

until the total number of variables to be analyzed has been studied and a point X' presenting the best functional value of the round is reached.

4. Pattern Move. If the functional value f' at the end of step 3 is better than the initial one f then the pattern move is tried. The coordinates of the f' point are incremented by an amount proportional to the change experienced for the coordinates in going from f to f'. This rate of change will be greater than one. If point f'', after the initial pattern move, happens to be better than f', a new step of length $(\lambda P)(\Delta P)$ is taken in the same direction. The role of λP here is identical to that of λ in the one-at-a-time portion of the process. The process here follows the same scheme explained in phase 3. As before, when a point is reached where no improvement is obtained by moving the vector either (ΔP) or $-(\Delta P)$, this point is considered the best of this series of pattern moves.

If the point obtained after a series of pattern moves is better than the point at the beginning of the series (i.e., at the end of the one-at-a-time round), a new round of the one-variable-at-a-time phase, as it was previously described, is attempted, and the process is kept going until no better points are found. If the pattern move phase happens to be a failure, a one-at-a-time round will be tried, resulting either in the final point, i.e., the optimum searched (as far as the technique can tell), or in the continuation of the optimization calculation.

It is easily seen from the above comments that the end point of the process will always be the starting point of a one-variable-at-a-time phase.

THE SAMPLE PROBLEMS

The techniques described previously are tested with a group of five two-variable unconstrained functions.

Three of the problems are by Witte and Holst[20]; we keep the names given to these functions in the original paper so they will be called: SHALOW, STRAIT, and CUBE. The fourth problem was presented for the first time by H. H. Rosenbrock[21] and also included by Witte and Holst who called it ROSIE. Our fifth problem is one presented and analyzed by E. M. L. Beale[22] and by Shah et al.[23]; we call this one BEALE.

The following are the algebraic expressions of our set of problems (to be minimized):

$$\text{ROSIE} = 100\ (y-x^2)^2 + (1-x)^2$$

$$\text{SHALOW} = (y-x^2)^2 + (1-x)^2$$

$$\text{STRAIT} = (y-x^2)^2 + 100(1-x)^2$$

$$\text{CUBE} = 100\ (y-x^3)^2 + (1-x)^2$$

$$\text{BEALE} = \sum_{i=1}^{3} U_i^2 \quad \text{where } U_i \equiv C_i - x(1-y^i)$$

and where $C_1 = 1.5$, $C_2 = 2.25$, $C_3 = 2.625$

---

[20]See reference 1, p. 1.

[21]Rosenbrock, H. H., "An Automatic Method for Finding the Greatest or Least Value of a Function," Computer J., Vol. 3, October 1960, 175-184.

[22]See reference 2, p. 1.

[23]See reference 14, p. 8.

ROSIE has a minimum $f = 0$ at $(1, 1)$, with a steep valley along $y = x^2$, and a side valley along the negative y - axis.

SHALOW presents a minimum of $f = 0$ at $(1, 1)$ with valleys along $y = x^2$ and $x = 1$. "SHALOW is similar to the function ROSIE but has a shallow valley compared with the steep valley of ROSIE."[24]

STRAIT has its minimum of $f = 0$ at $(1, 1)$ with a steep valley along $x = 1$.

CUBE presents a minimum $f = 0$ at $(1, 1)$ with a steep valley along $y = x^3$.

BEALE has a minimum of $f = 0$ at $(3, 0.5)$ with a narrow curving valley approaching the line $y = 1$.

Each one of the problems is solved beginning the optimization calculation at five different starting points so as to expose each procedure to a variety of topographical conditions. We use in ROSIE, SHALOW, STRAIT and CUBE the same initial points of Witte and Holst. We pick up for BEALE five of the starting values used by Shah et al., in fact the ones we feel to be the most difficult ones.

The results are recorded in the Tables of Appendix A. Each table contains, for a particular problem, the initial values together with the following information pertaining to each one of the optimizing codes:

a. Final Values. The optimum functional value together with the corresponding vector.

b. Number of times the evaluating function is called. In some of the codes this subroutine is called to evaluate the derivatives at some point without functional evaluation at all; however, these two calls

---

[24] See reference 1, page 1.

are not separated and both are recorded as functional evaluations.

c. Execution time in seconds. Internal clock readings are taken both at the beginning and end of each one of the problems by means of a library subroutine of the Berkeley System. This subroutine is for use on the BC 7090 equipped with the Delco clock on channel H.

d. Number of cycles. A cycle has a different meaning in each one of the codes. We describe very briefly the definition of cycle for the techniques we are dealing with:

BEST UNIVAR. A complete cycle includes the one-variable-at-a-time phase and the series of pattern moves following the previous one.

LOOK. A cycle is defined here as the exploratory search plus the pattern move.

VARMINT. A cycle includes establishing a direction to search, determining if the local minimum has been sufficiently well located and the modification of the H matrix on the bases of previous information.

MINFUN. A cycle here is as follows: determination of the gradient direction, step transverse to the gradient direction at the end of the previous step, prediction and verification of a minimum, step in the direction of the vector initial point - actual minimum.

STEP. Each iteration involves the necessary operations to locate a new restricted minimum.

ITERATED PARTAN. The cycle includes the operations (i) and (ii) explained in the description of this optimizing code.

CONTINUED PARTAN. A cycle here is understood as one including steps (i), (ii), (iii), and (iv) explained in the description of this optimizing procedure.

STEEPEST DESCENT. A cycle is defined here as the phase of the optimization process including steps (i), (ii), (iii), and (iv) of the code's description.

REMARK

The following general conclusions seem to be appropriate in view of the results summarized in Appendix A.

VARMINT presents the most consistent behavior among the group of techniques based on conventional mathematical methods. Similar results are reported by M. J. Box[25] working with a different group of techniques on a different set of problems. Box says that "Whilst these results are open to various interpretations, the conclusion reached is that Davidon's method is a more efficient optimizer than Rosenbrock's method, perhaps by a factor of 2 or 3."[26] Box's results together with our results are in accordance with the following remarks by Fletcher and Powell:[27] "Davidon's work has been little publicized, but in our opinion constitutes a considerable advance over current alternatives." Fletcher and Powell add in the same reference that their results with a variety of numerical tests "confirm that the method is probably the most powerful general procedure for finding a local minimum which is known at the present time."

STEP shows good results with functions ROSIE, SHALOW, STRAIT and CUBE. It does not work too well with BEALE, specially approaching the narrow

---

[25] Box, J. M., "Lecture Notes on Direct Search Methods of Optimization," Imperial Chemical Industries, Ltd., Central Instrumental Laboratory, England, October 1963 (13 pages).

[26] The same as ours.

[27] Fletcher, R., and M. J. D. Powell, "A Rapidly Convergent Descent Method for Minimization," The Computer J., Vol. 6, No. 2, 1963, 163-168.

curving "valley" from the almost flat region at the lower right corner of the space.

BEST UNIVAR of the direct search type of techniques exhibits the most consistent behavior of its group. LOOK requires lower computing time and presents better numerical values when it approaches the true optimum; however, its work on CUBE and BEALE is quite unsatisfactory.

One of the essential characteristics of direct search procedures for optimization is that they do not require derivatives of the objective function. These techniques are then adequate to treat difficult optimization problems involving functions with many variables. Furthermore, algebraic expressions for the optimization function are not necessary. All that is required is a way of finding functional values but without having the function available in algebraic form at all. In cases involving difficult functions or having no algebraic expressions for the objective function, the application of VARMINT or similar procedures would be impossible.

It is interesting to note the similarities in behavior between STEP and LOOK while optimizing BEALE. Further analysis of these results will be attempted in the near future.

CUBE proves to be the most difficult function for all the techniques. It takes the longest computing time to reach its true solution if compared against the other functions. MINFUN, ITERATED PARTAN, STEEPEST DESCENT and LOOK do not even approach the optimum of CUBE in most of the cases.

Some experiences with the same group of codes increasing the dimensionality of the function to be optimized and also introducing constraints will be reported elsewhere.

APPENDIX A

SUMMARY OF RESULTS

Table No. 1 ROSIE

Initial Values $x_o = -1.20000$ $y_o = 1.00000$ $F_o = 24.199997$

TECHNIQUE	FINAL VALUES			FUNCTIONAL EVALUATIONS	CYCLES	TIME
	x	y	F			
BEST UNIVAR	.99888	.99974	0.37E-07	875	31	1.85
LOOK	1.00000	.99999	0.63E-09	211	62	0.74
VARMINT	.99999	.99999	0.34E-10	37	35	1.77
MINFUN	1.00000	1.00000	0.22E-15	364	101	3.30
STEP	.99999	.99999	0.27E-12		500	0.88
ITERAT. PARTAN	0.58949	.57195	0.17E-00	786	100	4.08
CONT. PARTAN	.93911	.88165	0.37E-02	94	45	0.88
STEEPEST DESCENT	.98323	.96637	0.30E-03	202	100	1.91

Table No. 1 (cont.) ROSIE

Initial Values $x_o = -2.00000$ $y_o = -2.00000$ $F_o = 3{,}609.00000$

TECHNIQUE	FINAL VALUES			FUNCTIONAL EVALUATIONS	CYCLES	TIME
	x	y	F			
BEST UNIVAR	1.00027	1.00055	0.10E-06	861	29	1.75
LOOK	.99699	.99399	0.90E-05	188	42	0.61
VARMINT	.99999	.99999	0.65E-10	28	26	1.14
MINFUN	1.00000	1.00000	0.11E-10	303	90	2.07
STEP	1.00000	1.00000	0.43E-11		1800	3.02
ITERAT. PARTAN	.96117	.92294	0.16E-02	727	100	3.57
CONT. PARTAN	1.00155	1.00325	0.40E-05	139	20	0.62
STEEPEST DESCENT	.96519	.93147	0.12E-02	209	100	1.99

Table No. 1 (cont.) ROSIE

Initial Values $x_o = 5.62100$ $y_o = -3.63500$ $F_o = 124,141.15919$

TECHNIQUE	FINAL VALUES			FUNCTIONAL EVALUATIONS	CYCLES	TIME
	x	y	F			
BEST UNIVAR	1.00026	1.00055	0.11E-06	826	29	1.58
LOOK	1.00200	1.00400	0.40E-05	120	38	0.44
VARMINT	1.00000	1.00000	0.93E-12	35	33	1.38
MINFUN	.81290	.63490	0.10E-00	79	24	0.49
STEP	1.00000	1.00000	0.10E-11		1800	2.87
ITERAT. PARTAN	.87226	.75656	0.18E-01	795	100	4.44
CONT. PARTAN	1.00000	1.00000	0.96E-12	135	28	0.73
STEEPEST DESCENT	.85395	.72863	0.21E-01	278	100	1.91

Table No. 1 (cont.) ROSIE

Initial Values $x_o = -0.22100$ $y_o = 0.63900$ $F_o = 36.31961$

TECHNIQUE	FINAL VALUES			FUNCTIONAL EVALUATIONS	CYCLES	TIME
	x	y	F			
BEST UNIVAR	1.00010	1.00021	0.37E-07	877	28	1.61
LOOK	1.00499	1.01000	0.25E-04	226	55	0.68
VARMINT	1.00000	1.00000	0.75E-14	29	27	1.17
MINFUN	1.00000	1.00000	0.00E-00	290	77	2.61
STEP	.99999	.99999	0.17E-11		1200	1.62
ITERAT. PARTAN	.88191	.77395	0.15E-01	760	100	4.11
CONT. PARTAN	.93998	.88331	0.36E-02	100	45	0.88
STEEPEST DESCENT	.89278	.79377	0.13E-01	271	100	1.91

Table No. 1 (cont.) ROSIE

Initial Values $x_o = -2.54700$, $y_o = 1.48900$, $F_o = 2{,}510.79037$

TECHNIQUE	FINAL VALUES			FUNCTIONAL EVALUATIONS	CYCLES	TIME
	x	y	F			
BEST UNIVAR	1.00015	1.00032	0.52E-07	927	38	2.06
LOOK	-1.54390	2.38999	0.64E+01	49	6	0.38
VARMINT	.99998	.99997	0.43E-09	47	45	1.77
MINFUN	.99999	.99999	0.38E-10	268	83	1.28
STEP	1.00000	.99999	0.14E-11		700	1.10
ITERAT. PARTAN	.99999	.99999	0.18E-09	123	27	0.81
CONT. PARTAN	.99998	.99993	0.79E-07	115	40	0.83
STEEPEST DESCENT	-1.15155	1.35315	0.46E+01	294	100	1.91

Table No. 2 SHALOW

Initial Values $x_o = -2.00000$ $y_o = -2.00000$ $F_o = 45.00000$

TECHNIQUE	FINAL VALUES			FUNCTIONAL EVALUATIONS	CYCLES	TIME
	x	y	F			
BEST UNIVAR	1.00110	1.00250	0.13E-05	383	6	1.05
LOOK	1.00020	1.00040	0.39E-07	143	41	0.55
VARMINT	1.00000	1.00000	0.14E-12	10	8	0.90
MINFUN	.99983	.99959	0.34E-07	103	29	0.93
STEP	1.00000	1.00000	0.72E-15		2000	3.15
ITERAT. PARTAN	1.00034	1.00141	0.10E-05	425	40	1.72
CONT. PARTAN	.99941	.99975	0.10E-05	86	20	0.60
STEEPEST DESCENT	.99793	.99516	0.51E-05	80	40	0.79

Table No. 2 (cont.) SHALOW

Initial Values $x_o = 1.18400$ $y_o = 0.57400$ $F_o = 0.72000$

TECHNIQUE	FINAL VALUES			FUNCTIONAL EVALUATIONS	CYCLES	TIME
	x	y	F			
BEST UNIVAR	.99980	.99980	0.81E-07	182	4	0.71
LOOK	1.00000	1.00000	0.33E-12	145	39	0.36
VARMINT	1.00000	1.00000	0.00E-00	8	6	0.58
MINFUN	1.00020	1.00050	0.46E-07	108	33	1.09
STEP	1.00000	1.00000	0.00E-00		300	0.41
ITERAT. PARTAN	1.00000	1.00000	0.22E-15	25	10	0.22
CONT. PARTAN	.99998	.99998	0.86E-09	40	10	0.32
STEEPEST DESCENT	.99990	.99978	0.10E-07	28	10	0.22

Table No. 2 (cont.) SHALOW

Initial Values $x_o = 0.80300$ $y_o = -0.25100$ $F_o = 0.84000$

TECHNIQUE	FINAL VALUES			FUNCTIONAL EVALUATIONS	CYCLES	TIME
	x	y	F			
BEST UNIVAR	.99888	.99974	0.37E-07	875	31	1.85
LOOK	1.00000	.99999	0.63E-09	211	62	0.74
VARMINT	.99999	.99999	0.34E-10	37	35	1.77
MINFUN	1.00000	1.00000	0.22E-15	364	101	3.30
STEP	.99999	.99999	0.27E-12		500	0.88
ITERAT. PARTAN	0.58949	.57195	0.17E-00	786	100	4.08
CONT. PARTAN	.93911	.88165	0.37E-02	94	45	0.88
STEEPEST DESCENT	.98323	.96637	0.30E-03	202	100	1.91

Table No. 2 (cont.) SHALOW

Initial Values $x_o = 0.21100$ $y_o = 3.50500$ $F_o = 12.60000$

TECHNIQUE	FINAL VALUES			FUNCTIONAL EVALUATIONS	CYCLES	TIME
	x	y	F			
BEST UNIVAR	.99910	.99789	0.91E-06	349	6	0.92
LOOK	1.00020	1.00040	0.39E-07	135	37	0.44
VARMINT	1.00000	1.00000	0.15E-13	9	7	0.53
MINFUN	1.00000	1.00000	0.00E-00	271	68	3.28
STEP	1.00000	1.00000	0.29E-14		400	0.52
ITERAT. PARTAN	1.00012	1.00033	0.22E-07	22	10	0.22
CONT. PARTAN	1.00000	1.00000	0.22E-15	93	20	0.47
STEEPEST DESCENT	1.05617	1.15377	0.46E-02	79	40	0.79

Table No. 2 (cont.) SHALOW

Initial Values $x_o = 0.82000$, $y_o = 4.69000$, $F_o = 16.17000$

TECHNIQUE	FINAL VALUES			FUNCTIONAL EVALUATIONS	CYCLES	TIME
	x	y	F			
BEST UNIVAR	.99969	.99899	0.25E-06	389	6	0.97
LOOK	.99999	.99999	0.10E-11	93	26	0.34
VARMINT	1.00000	1.00000	0.58E-13	10	8	0.57
MINFUN	1.00000	1.00000	0.00E-00	338	103	2.72
STEP	1.00000	1.00000	0.17E-11		200	0.32
ITERAT. PARTAN	.99966	.99918	0.13E-06	83	20	0.53
CONT. PARTAN	.99999	1.00000	0.93E-12	159	25	0.66
STEEPEST DESCENT	1.61882	2.87419	0.44E-00	79	40	0.79

Table No. 3 STRAIT

Inital Values $x_o = 2.00000$ $y_o = -2.00000$ $F_o = 136.00000$

TECHNIQUE	FINAL VALUES			FUNCTIONAL EVALUATIONS	CYCLES	TIME
	x	y	F			
BEST UNIVAR	1.00000	.99999	0.18E-09	177	4	0.86
LOOK	1.00000	1.00000	0.22E-10	62	14	0.36
VARMINT	1.00000	1.00000	0.55E-11	6	4	0.78
MINFUN	1.00000	.99993	0.44E-08	110	32	1.22
STEP	1.00000	1.00000	0.56E-14		400	0.88
ITERAT. PARTAN	1.00000	1.00000	0.12E-11	19	10	0.21
CONT. PARTAN	1.00000	1.00001	0.47E-10	75	10	0.29
STEEPEST DESCENT	1.00000	.99999	0.14E-10	19	10	0.22

Table No. 3 (cont.) STRAIT

Initial Values $x_o = 2.00000$ $y_o = 2.32200$ $F_o = 139.96000$

TECHNIQUE	FINAL VALUES			FUNCTIONAL EVALUATIONS	CYCLES	TIME
	x	y	F			
BEST UNIVAR	1.00002	1.00023	0.72E-07	209	4	0.76
LOOK	.99999	1.00000	0.39E-10	104	21	0.37
VARMINT	1.00000	1.00000	0.83E-11	6	4	0.44
MINFUN	.99999	.99963	0.13E-07	78	33	0.48
STEP	1.00000	.99999	0.11E-10		200	0.38
ITERAT. PARTAN	1.00000	1.00000	0.27E-13	25	10	0.23
CONT. PARTAN	1.00000	1.00000	0.91E-13	52	10	0.29
STEEPEST DESCENT	1.00000	1.00000	0.37E-13	20	10	0.22

Table No. 3 (cont.) STRAIT

Initial Values $x_o = 2.01900$ $y_o = -1.50500$ $F_o = 134.99000$

TECHNIQUE	FINAL VALUES			FUNCTIONAL EVALUATIONS	CYCLES	TIME
	x	y	F			
BEST UNIVAR	1.00000	.99970	0.89E-07	213	6	0.76
LOOK	1.00000	1.00000	0.23E-10	119	17	0.34
VARMINT	1.00000	1.00000	0.43E-10	6	4	0.40
MINFUN	1.00000	.99998	0.30E-09	176	45	1.49
STEP	1.00000	1.00000	0.17E-13		300	0.58
ITERAT. PARTAN	1.00000	1.00000	0.10E-11	19	10	0.22
CONT. PARTAN	1.00000	1.00000	0.00E-00	50	10	0.30
STEEPEST DESCENT	1.00000	.99999	0.49E-11	19	10	0.22

Table No. 3 (cont.) STRAIT

Initial Values $x_o = 1.99200$ $y_o = -3.22200$ $F_o = 150.10000$

TECHNIQUE	FINAL VALUES			FUNCTIONAL EVALUATIONS	CYCLES	TIME
	x	y	F			
BEST UNIVAR	1.00000	1.00020	0.40E-07	222	4	0.77
LOOK	.99999	1.00000	0.56E-10	144	46	0.39
VARMINT	1.00000	1.00000	0.11E-11	8	6	0.61
MINFUN	1.00010	1.00320	0.98E-05	103	24	0.52
STEP	1.00000	1.00002	0.26E-09		200	0.37
ITERAT. PARTAN	1.00000	.99999	0.66E-12	20	10	0.22
CONT PARTAN	1.00000	1.00000	0.11E-11	52	10	0.29
STEEPEST DESCENT	1.00000	.99999	0.19E-11	19	10	0.22

Table No. 3 (cont.) STRAIT

Initial Values $x_o = 1.98600$, $y_o = 5.22700$, $F_o = 98.86000$

TECHNIQUE	FINAL VALUES			FUNCTIONAL EVALUATIONS	CYCLES	TIME
	x	y	F			
BEST UNIVAR	1.00000	1.00000	0.68E-10	223	6	0.77
LOOK	.99999	.99999	0.51E-10	129	32	0.37
VARMINT	1.00000	1.00000	0.27E-09	6	4	0.40
MINFUN	1.00000	1.00310	0.93E-05	82	24	0.49
STEP	1.00000	1.00000	0.22E-11		300	0.56
ITERAT. PARTAN	1.00000	1.00000	0.35E-10	20	10	0.22
CONT. PARTAN	1.00000	1.00001	0.10E-11	21	10	0.22
STEEPEST DESCENT	1.00000	1.00000	0.20E-14	25	10	0.23

Table No. 4 CUBE

Initial Values $x_o = -1.20000$ $y_o = 1.00000$ $F_o = 24.199997$

TECHNIQUE	FINAL VALUES			FUNCTIONAL EVALUATIONS	CYCLES	TIME
	x	y	F			
BEST UNIVAR	.98770	.96349	0.15E-03	1049	21	2.08
LOOK	1.00990	1.03000	0.98E-04	191	41	0.52
VARMINT	1.00000	1.00000	0.23E-12	34	31	1.69
MINFUN	.12966	-0.00947	1.69E+00	121	11	0.58
STEP	1.00000	1.00000	0.21E-12		1500	1.86
ITERAT. PARTAN	.63960	.25778	0.13E+00	802	100	4.07
CONT. PARTAN	.99950	.99858	0.98E-06	300	70	1.64
STEEPEST DESCENT	-1.06003	-1.19364	0.42+01	297	100	1.91

Table No. 4 (cont.) CUBE

Initial Values $x_o = 1.39100$ $y_o = -2.60600$ $F_o = 2,806.40000$

TECHNIQUE	FINAL VALUES			FUNCTIONAL EVALUATIONS	CYCLES	TIME
	x	y	F			
BEST UNIVAR	1.00475	1.00143	0.22E-04	1979	36	3.20
LOOK	1.00100	1.00300	0.99E-06	223	47	0.58
VARMINT	.99999	.99997	0.13E-09	50	48	1.86
MINFUN	.83325	.05787	2.77E-02	137	22	0.62
STEP	.99999	.99999	0.75E-12		1700	2.47
ITERAT. PARTAN	-1.17301	-1.62346	0.47E+01	331	100	1.91
CONT. PARTAN	.99997	.99989	0.12E-08	498	100	2.43
STEEPEST DESCENT	-1.20967	-1.77292	0.48E+01	297	100	1.91

Table No. 4 (cont.) CUBE

Initial Values $x_o = 1.24300$ $y_o = -1.97400$ $F_o = 1{,}516.80000$

TECHNIQUE	FINAL VALUES			FUNCTIONAL EVALUATIONS	CYCLES	TIME
	x	y	F			
BEST UNIVAR	1.00191	1.00575	0.36E-05	1774	33	2.92
LOCK	1.00200	1.00600	0.40E-05	211	44	0.56
VARMINT	1.00000	1.00000	0.27E-07	20	18	0.91
MINFUN	1.00000	1.00000	0.50E-11	322	95	1.91
STEP	1.00000	.99999	0.89E-13		2000	3.28
ITERAT. PARTAN	-0.95409	-0.87631	0.38E+01	1196	100	4.39
CONT. PARTAN	.99895	.99685	0.11E-05	321	70	1.82
STEEPEST DESCENT	-1.02081	-1.06595	0.41E+01	297	100	1.91

Table No. 4 (cont.) CUBE

Initial Values $x_o = 0.24800$ $F_o = 964.54000$

$y_o = -3.08200$

TECHNIQUE	FINAL VALUES			FUNCTIONAL EVALUATIONS	CYCLES	TIME
	x	y	F			
BEST UNIVAR	1.00062	1.00188	0.40E-06	2092	35	3.27
LOOK	-1.11590	-1.39190	0.45E+01	57	8	0.27
VARMINT	1.00000	1.00000	0.56E-12	43	40	1.63
MINFUN	0.58413	.19943	1.73E-01	134	25	0.60
STEP	.99990	.99970	0.10E-07		1000	1.59
ITERAT. PARTAN	.64664	.26671	0.12E+00	816	100	4.49
CONT. PARTAN	.99999	.99999	0.99E-12	307	70	2.06
STEEPEST DESCENT	-0.96120	-0.88913	0.34E+01	298	100	1.91

Table No. 4 (cont.) CUBE

Initial Values $x_o = -1.20000$ $F_o = 57.84000$

$y_o = -1.00000$

TECHNIQUE	FINAL VALUES			FUNCTIONAL EVALUATIONS	CYCLES	TIME
	x	y	F			
BEST UNIVAR	.98770	.96350	0.15E-03	1379	26	2.39
LOOK	-0.99990	-1.00000	0.40E+01	34	3	0.23
VARMINT	1.00000	1.00000	0.92E-12	47	45	1.77
MINFUN	.10641	.00006	8.00E-01	38	11	0.41
STEP	1.00000	1.00000	0.52E-13		1000	1.72
ITERAT. PARTAN	.72142	.37316	0.78E-01	858	100	4.12
CONT. PARTAN	1.00000	1.00000	0.11E-12	326	75	2.04
STEEPEST DESCENT	-0.96120	-0.88913	0.34E+01	298	100	1.91

Table No. 5 BEALE

Initial Values $x_o = 0.00000$ $y_o = 0.00000$ $F_o = 14.20000$

Technique	FINAL VALUES			FUNCTIONAL EVALUATIONS	CYCLES	TIME
	x	y	F			
BEST UNIVAR	3.00000	.50000	0.28E-11	156	5	0.68
LOOK	.70010	.70010	0.89E+01	40	4	0.36
VARMINT	3.00000	.50000	0.18E-14	11	9	0.95
MINFUN	3.00000	.50000	0.11E-14	314	68	2.60
STEP	3.00000	.50000	0.23E-11		300	0.65
ITERAT. PARTAN	2.99983	.49993	0.20E-07	125	25	0.79
CONT. PARTAN	2.99869	.49962	0.33E-06	201	100	1.90
STEEPEST DESCENT	2.97795	.49438	0.80E-04	199	100	1.90

Table No. 5 (cont.) BEALE

Initial Values $x_o = 8.00000$ $y_o = 0.20000$ $F_o = 81.70000$

TECHNIQUE	FINAL VALUES			FUNCTIONAL EVALUATIONS	CYCLES	TIME
	x	y	F			
BEST UNIVAR	3.00224	.50060	0.84E-06	298	7	1.08
LOOK	8.80010	1.00010	0.75E+00	50	5	0.32
VARMINT	3.00000	.500000	0.10E-10	29	27	1.19
MINFUN	3.08110	.51885	0.97E-04	234	101	0.88
STEP	2.04505	.07556	0.54E+00		1100	2.15
ITERAT. PARTAN	2.99999	.49999	0.64E-12	353	100	2.27
CONT. PARTAN	3.00234	.50059	0.87E-06	78	35	0.70
STEEPEST DESCENT	7.83932	.02590	0.25E+00	199	100	1.91

Table No. 5 (cont.) BEALE

Initial Values $x_o = 5.00000$ $y_o = 0.80000$ $F_o = 0.49000$

TECHNIQUE	FINAL VALUES			FUNCTIONAL EVALUATIONS	CYCLES	TIME
	x	y	F			
BEST UNIVAR	3.00052	.50014	0.48E-07	528	13	1.56
LOOK	5.10010	.90010	0.40E+00	26	1	0.24
VARMINT	3.00000	.50000	0.22E-12	16	14	0.75
MINFUN	3.00000	.50000	0.12E-11	153	45	1.11
STEP	3.00000	.50000	0.36E-13		1200	1.66
ITERAT. PARTAN	3.00332	.50101	0.26E-05	461	75	2.04
CONT. PARTAN	3.00000	.50000	0.37E-12	83	35	0.76
STEEPEST DESCENT	4.95171	.75065	0.13E+00	199	100	1.91

Table No. 5 (cont.) BEALE

Initial Values $x_o = 8.00000$ $y_o = 0.80000$ $F_o = 2.04200$

TECHNIQUE	FINAL VALUES			FUNCTIONAL EVALUATIONS	CYCLES	TIME
	x	y	F			
BEST UNIVAR	2.99880	.49974	0.26E-06	595	14	1.70
LOOK	8.20010	1.00010	0.11E+01	34	2	0.26
VARMINT	3.00000	.50000	0.28E-11	26	24	1.09
MINFUN	3.00000	.50000	0.39E-11	276	88	1.60
STEP	7.99613	.87417	0.38E+00		100	0.18
ITERAT. PARTAN	2.99982	.49995	0.61E-08	377	75	2.07
CONT. PARTAN	3.00000	.50000	0.00E+00	64	30	0.65
STEEPEST DESCENT	7.99256	.85937	0.25E+00	199	100	1.91

Table No. 5 (cont.) BEALE

Initial Values $x_o = 2.00000$ $y_o = 0.80000$ $F_o = 6.27000$

TECHNIQUE	FINAL VALUES			FUNCTIONAL EVALUATIONS	CYCLES	TIME
	x	y	F			
BEST UNIVAR	2.99789	.49950	0.72E-06	359	8	1.21
LOOK	2.10010	.90010	0.59E+01	26	1	0.23
VARMINT	3.00000	.50000	0.29E-11	10	8	0.56
MINFUN	3.00000	.49999	0.70E-10	138	40	0.98
STEP	3.00000	.50000	0.49E-12		500	1.05
ITERAT. PARTAN	2.99956	.49984	0.94E-07	159	25	0.72
CONT. PARTAN	3.00000	.50000	0.22E-15	87	40	0.84
STEEPEST DESCENT	2.98857	.49734	0.21E-04	200	100	1.91

APPENDIX B

FIGURES

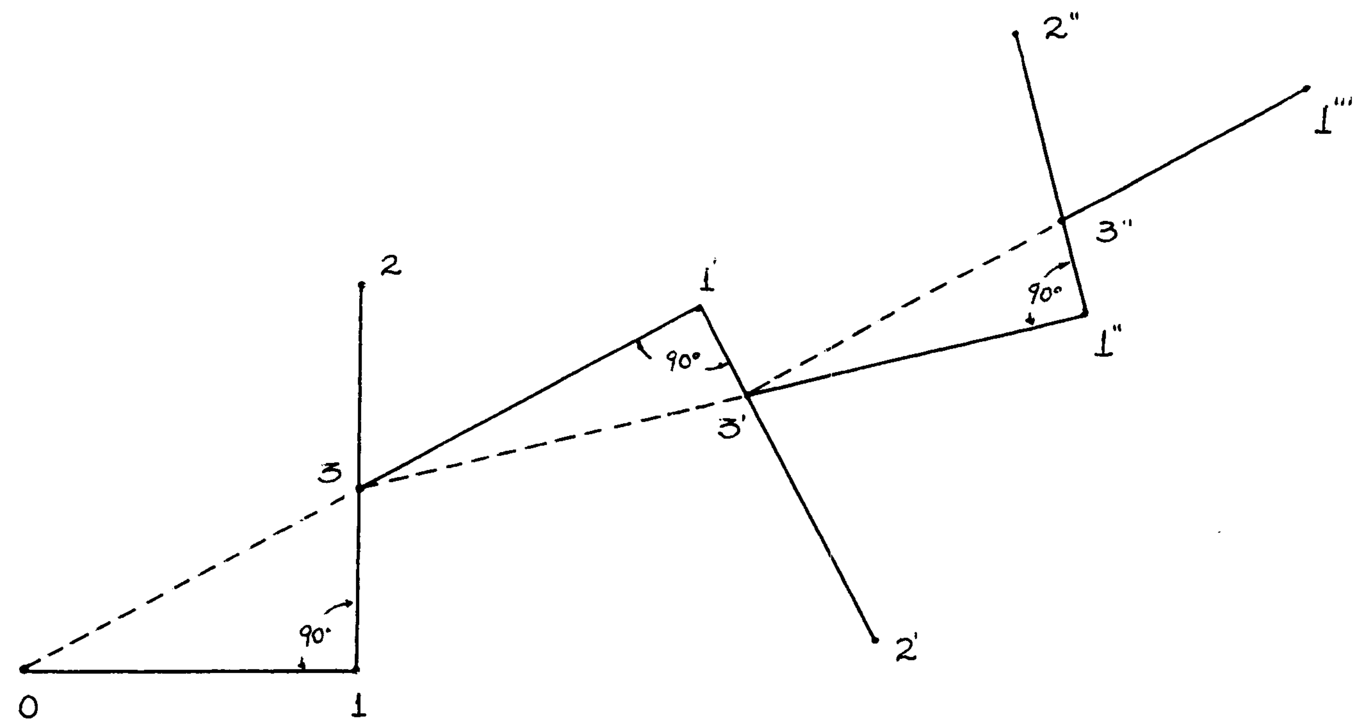

FIGURE No. 1

GENERAL MINIMIZING ROUTINE

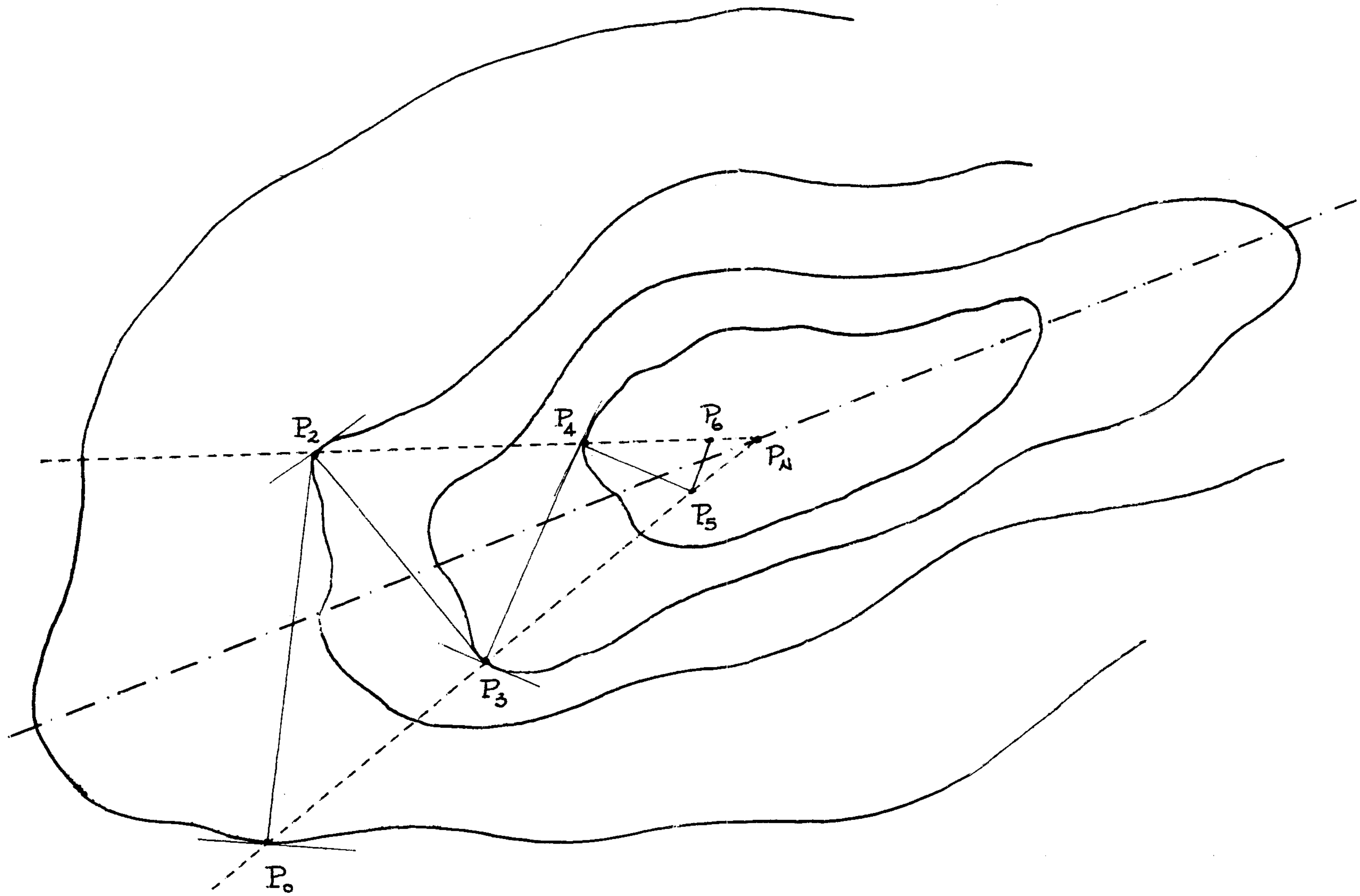

FIGURE No. 2

STEEPEST DESCENT METHOD

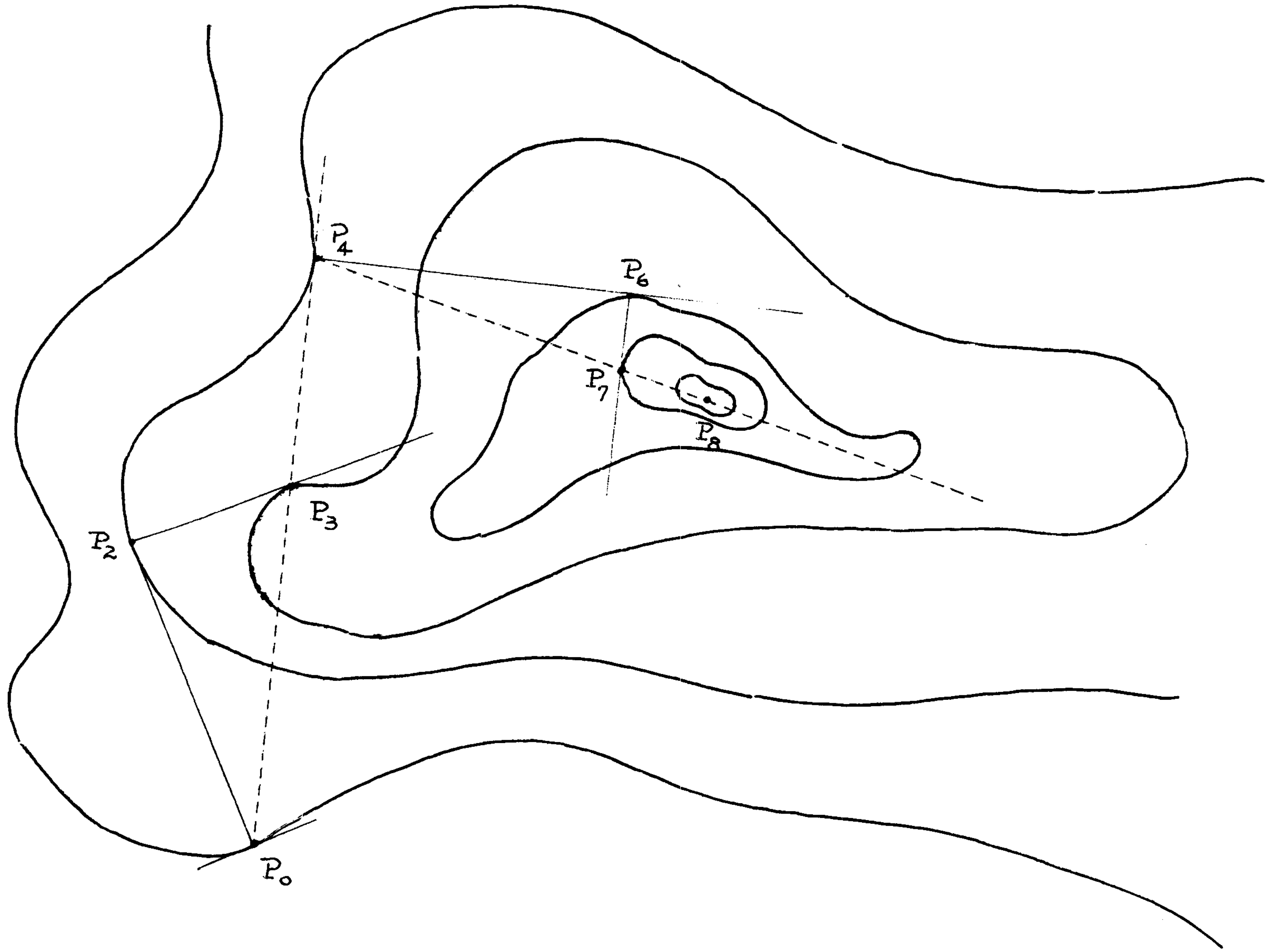

FIGURE No. 3

ITERATED PARTAN

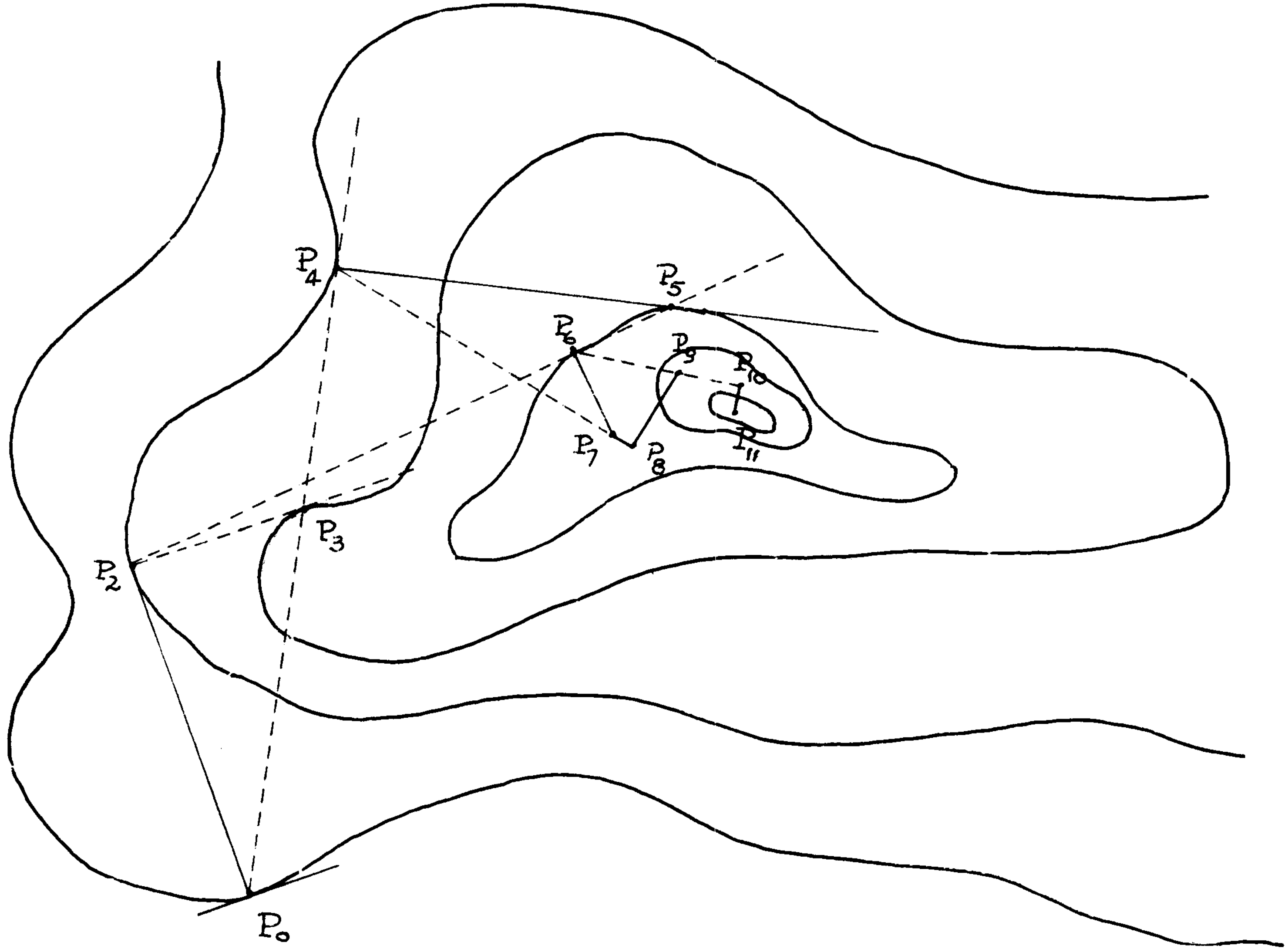

FIGURE No. 4

CONTINUED PARTAN